Machine Learning for Calligraphy Styles Recognition

Chen, Yu-Sheng
Institute for Computational and Mathematical Engineering,
Stanford University
yusheng@stanford.edu

Su, Guangjun
Department of Management Science and Engineering,
Stanford University
gsu2@stanford.edu

Li, Haihong
Department of Electrical Engineering,
Stanford University
hhli@stanford.edu

***Abstract*— Calligraphy is a visual art related to writing. Different calligraphy styles render Chinese characters, which are already structurally complicated, in different ways and thus making recognizing the styles even more challenging for computers. This paper discusses several machine learning algorithms applied to mainstream Chinese calligraphy styles recognition, and investigates their effectiveness.**

I. INTRODUCTION

Chinese calligraphy is a distinguished part in Chinese traditional art and culture. There is a general standardization of various calligraphy styles in tradition, such as seal script, clerical script, regular script, running script, cursive script, among others, as is shown in Fig. 1. For each script, there are a handful of notable calligraphers in history, whose styles vary albeit there are some degree of standardization for the script.

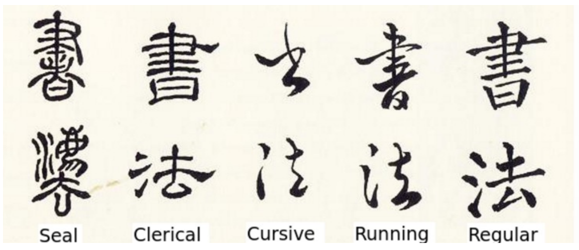

Fig. 1. The word "calligraphy" in five calligraphy styles

The input is thousands of images, each containing a single character written in a calligraphy style, and the output is the predicted label of the character's calligraphy styles.

Softmax Regression, Support Vector Machine (SVM), k-Nearest Neighbors (kNN), Random Forests are investigated. In the process of hyper-parameter tuning, the technique k-fold cross validation is used to produce the nearly optimal model for each learning algorithm. In the process of feature extraction, Histogram of Oriented Gradients descriptor (HOG) are used. In the process of picking the best algorithm, hold-out cross validation is applied to the optimal models generated from these learning algorithms. This paper uses training/testing accuracy and confusion matrix covariance to measure the performance of algorithms. TensorFlow and OpenCV are utilized.

II. RELATED WORK

In machine learning applications, there has been some character recognition works published in the field of multinational handwriting texts such as Western characters, numeral characters and Chinese calligraphy. And Chinese calligraphy, especially the traditional writing, is considered complicated for style recognition because the characters are composed of multiple strokes following different script styles. And different calligraphy artists blend in different techniques and characteristics under different emotions and personal preference, which makes style recognition very challenging. Existing researches on digital Chinese calligraphy processing in general can be primarily divided into three classes: (1). character recognition and retrieval [1][2], (2). calligraphy simulation and generation [3][4] and (3). calligraphy analysis. For the applications such as reprinting documents and optical character recognition, character recognition and retrieval are core parts in those processes. And calligraphy simulation and generation focus on creating artistic writing based on another text source. Other works such as writer identification, style recognition, and calligraphy evaluation [5][6][7] are grouped into the last class. Since the Chinese calligraphy style plays a key role in calligraphy training, appreciation and retrieval, in this project, we propose machine learning models for style recognition of traditional Chinese calligraphy.

III. DATASETS AND FEATURES

The dataset has 5 different script styles. There are regular style, clerical style, seal style, cursive style, and running style, each labeled 1 to 5 respectively. There are over 2000 multicolored images of Chinese characters for each style, each containing a single character. We split the dataset into training set and testing set, as indicated in Table I.

TABLE I
THE DATASET

Label	Style	Training set	Testing set
1	Regular	1500	505
2	Clerical	1500	500
3	Seal	1500	500
4	Running	1500	514
5	Cursive	1500	500

Intuitively, as a gross simplification, the regular style is the canonical way of writing. It faithfully renders the true structure of a character, without much distortion, stoke concatenation or omission. The clerical style tends to prolong the character horizontally and compress vertically. The seal style, usually used in seal craving, is more compact and distorted. The running style is featured by its rounded, concatenated strokes mild simplification of the character. The cursive style, however, is much of aesthetic taste and less of usefulness in everyday life, due to its oversimplification.

As for feature extraction, the first approach is downsampling and pixel mapping. Initially, each image has 370*370 pixels, so they need to be downsized to 28*28 grayscale images to speed up the training process. After downsampling, each images pixel values are read in and mapped into a vector, a simple but generally effective image feature mapping. Another approach for feature extraction is downsampling and Histogram of oriented gradients (HOG). HOG is widely used for image feature extraction for object detection in computer vision fields. It has also been proved that gradient descriptors can effectively improve the accuracy of machine learning algorithms.

IV. METHODS

Softmax Regression, Support Vector Machine (SVM), k-Nearest Neighbors (kNN), Random Forests are investigated.

A. Softmax Regression

Softmax Regression is a classic learning algorithm used to model a multinomial distribution. In our problem, there are 5 classes, thus the target variable is $y = \{1, 2, 3, 4, 5\}$. The log-likelihood to be maximized using stochastic gradient descent is $l(\theta) = \sum_{i=1}^{m} \log \prod_{l=1}^{5} \left(\frac{e^{\theta_l^T x^{(i)}}}{\sum_{j=1}^{5} e^{\theta_j^T x^{(i)}}} \right)^{1\{y^{(i)}=l\}}$, where m is the number of training examples and θ is the parameter associating features x with the target variable y in $y = \theta^T x$. The predicted label is the outcome with the highest probability $P(y = i) = \frac{e^{\theta_i^T x^{(i)}}}{\sum_{j=1}^{5} e^{\theta_j^T x^{(i)}}}$.

B. Support Vector Machine

Support Vector Machine (SVM) model can be expressed as the following optimization problem, $min_{\gamma,\omega,b}\frac{1}{2}||\omega||^2 + C\sum_{i=1}^{m}\xi_i$ such that $y^{(i)}(\omega^T x^{(i)} + b) \geq 1 - \xi_i$, for all $i = 1, 2, ..., m.$ and all ξ_i are non-negative. Introducing ξ_i can make SVM less sensitive to the outliers, while the term $C\sum_{i=1}^{m}\xi_i$ actually implies the cost we would pay to the objective function. In our approach, we use Gaussian radial basis function (RBF) as our kernel, which has the form $K(x^{(i)}, x^{(j)}) = \exp(-\gamma||x^(i) - x^(j)||^2)$, for $\gamma > 0$. The hyper-parameters C, γ are determined by grid search using cross-validation technique.

C. k-Nearest Neighbors

In pattern recognition, the k-Nearest Neighbors algorithm (kNN) is a non-parametric classification method. An object is classified by a majority vote of its neighbors priorly stored in the training stage, with the object being assigned to the class which is most common among its k nearest neighbors (k is a positive integer, typically small). For example, if $k = 1$, then the object is simply assigned to the class of that single nearest neighbor.

D. Random Forests

Random Forest is an ensemble method for classification. The key idea is to construct a brunch of weak learner which can come together as a strong learner. In this algorithm, the weak learner is a decision tree. To construct a decision tree, we need to choose a variable to "best" split the data set at each depth, and Gini impurity and information gain are the two common metrics to tell which is the best. In our Random Forest model, we use information gain as the splitting metrics. In our problem, we use ID3 algorithm to generate decision trees, and at each iteration we need to find a best splitting variable to maximize the information gain, which is $IG(T,a) = H(T) - H(T|a)$, where T denotes a set of training samples, $H(T) = -\sum_{i=1}^{k} p_i \log p_i$, and $H(T|a) = \sum_{v \in vals(a)} \frac{|\{x \in T | x_a = v\}|}{|T|} H(\{x \in T | x_a = v\})$ is the weighted sum of entropy of the children after splitting at variable x_a. And then we use a bagging methodology to train our random forest, which is to train each decision tree based on a subset of the total features, and finally use the majority vote to integrate the output of each decision tree. The process of building a random forest model is shown in Fig. 2.

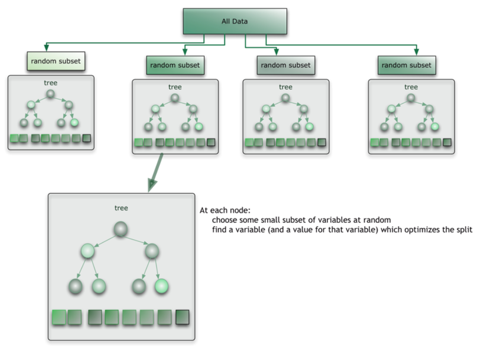
All Data
random subset
random subset
random subset
random subset
tree
tree
tree
tree
tree
At each node:
choose some small subset of variables at random
find a variable (and a value for that variable) which optimizes the split

Fig. 2. Building a random forest

E. Convolutional Neural Networks

In machine learning, a convolutional neural network (CNN) is a type of feed-forward artificial neural network in which the connectivity pattern between its neurons is inspired by the organization of the animal visual cortex. The response of an individual neuron to stimuli within its receptive field can be approximated mathematically by a convolution operation. Nowadays CNN has wide applications in image and video recognition and natural language processing. And it is also famous for its shift invariant or space invariant characteristics[8].

V. Experiments and Results

The training set is fed into these learning algorithms and generate an optimal model for each algorithms. To produce the optimal model, or practically speaking, the nearly optimal model, for each learning algorithm, we use k-fold cross validation to tune the hyper-parameters. After that we use the testing set to evaluate the optimal models generated by these learning algorithms.

Apart from using the testing accuracy as a metric the measure the models' performance, we propose to use confusion matrix covariance as well. Each column represents the instances in a predicted class, and each row represents the instances in an actual class. The more similar the confusion matrix is to a diagonal matrix, the more precise the classification model is. The diagonality of the confusion matrix can be measured by the covariance between the two indicies of the matrix $Cov(X,Y) = E((X - E(X))(Y - E(Y))^T)$. It yields 1 for perfect diagonal matrix and -1 for perfect anti-diagonal matrix, and it yields $\sim$0 for a random matrix. This metric is robust: it is unchanged if one scale the matrix. Clearly, the confusion matrix of a perfect classification model should be a diagonal matrix, and its covariance should be 1.

A. Softmax Regression

The step size controls the convergence speed of stochastic gradient descent. If it is too small, the training may consume to much time and the algorithm is easy to fall into local optimum. If it is too large, however, the algorithm may oscillate around the optimum or failed converge. To avoid picking a optimal step size manually, the technique of adaptive step size is used. That is, the step size in the next iteration is determined by the difference of the parameters between this iteration and the previous iteration.

Paired with the HOG descriptor, Softmax Regression yielded the most satisfying result. The testing accuracy is 0.9555 and the training accuracy is 0.9680. The confusion covariance is 0.9415, which means the confusion matrix is nearly a diagonal matrix. Without the HOG descriptor, the accuracy is much lower but still better than random guessing. The visualization of the confusion matrix is shown in Fig. 3 and Fig. 4.

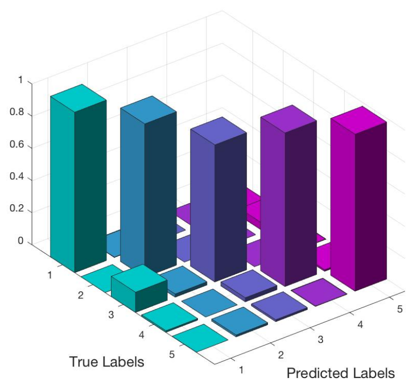

Fig. 3. Confusion matrix: Softmax Regression with HOG

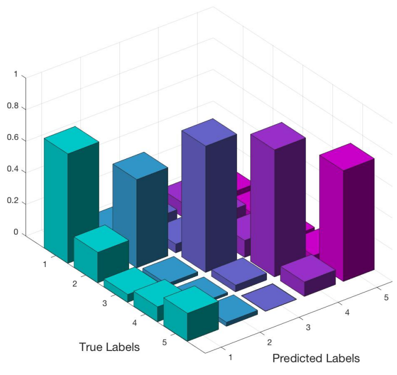

Fig. 4. Confusion matrix: Softmax Regression without HOG

B. *Support Vector Machine*

There are 2 hyper-parameters, C and γ, to be determined in SVM model. γ controls the RBF kernel bandwidth for our non-linear classification. And the regularization parameter, C, will be used to avoid over-fitting. To search for the best configuration of C and γ, we use 2-D grid search in log scale, with all data grid points spaced evenly on a log scale. For C we search the range $[2^0, 2^{10}]$, and search $[2^{-7}, 2^4]$ for γ. As the figure shown below, we can see the best γ value is around 2. However, the cross validation error in our data is not very sensitive to the value of C. Finally $C = 181.019$ and $\gamma = 2.378$ give us the best accuracy.

The testing accuracy of this Support Vector Machine is 0.8694 and the confusion matrix covariance is 0.8089. The visualization of the confusion matrix is shown in Fig. 5.

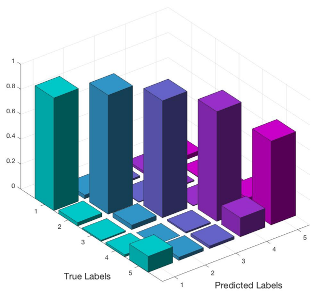

Fig. 5. Confusion matrix: Support Vector Machine with HOG

C. *k-Nearest Neighbors*

In kNN model, to optimize the performance, we have to find the best k value which minimizes the cross-validation error. In our experiment, we run the cross-validation process over $k = 1, 2, ..., 15$. As shown in Fig. 6, $k = 1$ yields the best cross-validation accuracy, however when we use $k = 1$ on test set, over-fitting actually degrades the test accuracy very much. By double checking with the final test accuracy, we find $k = 4$ actually fits the data best with the minimal test error and reasonable training error.

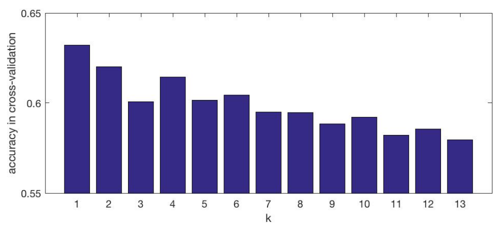

Fig. 6. kNN cross validation accuracy versus k

Using k=4, the testing accuracy of the model generated by kNN is 0.7511, and the confusion matrix covariance is 0.6385. The visualization of the confusion matrix is shown in Fig. 7.

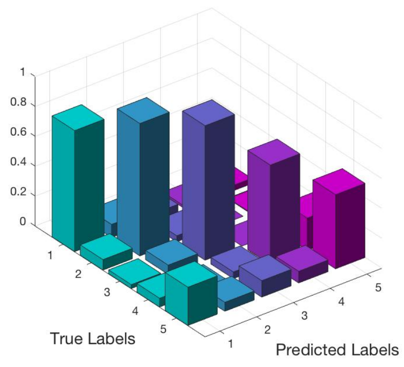

Fig. 7. Confusion matrix: k-Nearest Neighbors with HOG

D. *Random Forest*

In terms of Random Forest, there are a few parameters can be tuned, and the number of tree and the maximum depth of each tree are two of the most important parameters. In order to tune these two parameters, we use the 10-fold cross validation on the training set. The z-axis denotes the mean accuracy on the validation set, and the x-axis and y-axis denote the tree number and depth number respectively, shown as the following figure. The tree number ranges from 50 to 500 with 50 step size, and depth number ranges from 3 to 9 with 1 step size.

From Fig. 8, we know that as the depth number increases, the mean accuracy on the validation set

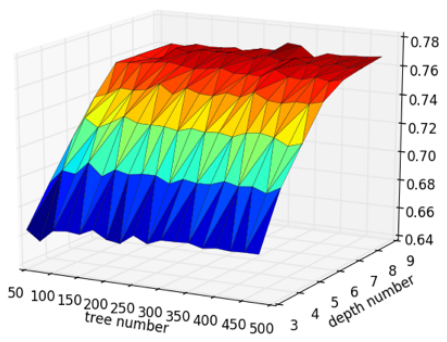

Fig. 8. Parameter tuning for Random Forest algorithm

increases as well, but when it becomes flat after the depth number is 7. However, as the depth number increase, the running time of the algorithm increases. In terms of accuracy and efficiency, we think 7 may be the best parameter for our model. Also from Fig. 8 we know that the tree number does not have a crucial influence on our problem. Commonly speaking, the more trees in the random forest, the more stable of the model. However, considering the efficiency of our model, we choose tree number as 250.

The testing accuracy of the model is 0.7852 and the confusion matrix covariance is 0.7356. The visualization of the confusion matrix is shown in Fig. 9.

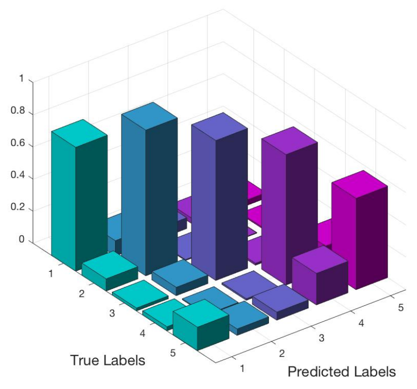

Fig. 9. Confusion matrix: Random Forest with HOG

E. Convolutional Neural Networks

For CNN part, we tried several configurations up to 5 layers (denoted as L5), due to computation resource limitations. In Li's work [9] in the CS231N project, he reported the best performance within L6 to L13 is L11, which yields the accuracy rate 88.6% on the same data set. As we can see in the accuracy summary table, softmax with HOG descriptor outperforms CNN's best results by a considerable amount. Regarding the super high computation requirement in CNN, we can conclude softmax classification is a more efficient and accurate model in this application, while CNN still guarantees competitive performance without using any extra powerful feature extraction techniques.

VI. Conclusion and Future Works

From the experiments above, we may conclude that:

- For this classification problem, Softmax Regression with HOG descriptor outperforms all other ML algorithms, including CNN and SVM. This algorithm stands out with its testing accuracy of 0.9555 and confusion matrix covariance of 0.9415.
- Softmax with HOG can even beat human judgment with respect to running and cursive styles.
- Traditional ML with relevant features can be more accurate and efficient than CNN, while CNN can do excellent jobs without designing advanced feature extraction methods (domain knowledge)
- Feature extraction is the key factor to boost learning accuracy. This is the key reason for the success of traditional algorithms.

There is still space for development. For example, the model could be trained to classify calligrapher artists' styles but more data is needed. Also, a more complicated CNN may be used to achieve a even higher accuracy.

References

[1] Lu W, Wu J, Wei B, Zhuang Y, Efficient Shape Matching for Chinese Calligraphic Character Retrieval, Zhejiang University Press, pp. 873-884, 2011.
[2] Yu K, Wu J, Zhuang Y, Skeleton-Based Recognition of Chinese Calligraphic Character Image, Advances in Multimedia Information Processing-PCM 2008, pp. 228-237, 2008.
[3] Leung H, Wong S T S, Ip H H S, In the Name of Art, IEEE Signal Processing Magazine, 25(4): 49-54, 2008.
[4] Xu S, Lau F C M, Cheung W K, Pan Y, Automatic Generation of Artistic Chinese Calligraphy, IEEE Intelligent Systems, 20(3): 32-39, 2005.
[5] Li X, Ding X, Writer Identification of Chinese Handwriting Using Grid Microstructure Feature, Advances in Biometrics, pp. 1230-1239, 2009.
[6] Zhuang Y, Lu W, Wu J, Latent Style Model: Discovering Writing Styles for Calligraphy Works, Journal Of Visual Communication And Image Representation, 20(2):84-96, 2009.
[7] Han C C, Chou C H, Wu C S, An Interactive Grading and Learning System for Chinese Calligraphy, Machine Vision And Applications, 19(1):43-55, 2008.
[8] "Convolutional Neural Networks (LeNet) DeepLearning 0.1 documentation".DeepLearning 0.1. LISA Lab. Retrieved 31 August 2013.
[9] Boqi Li, CS231N final project. Convolution Neural Network for Traditional Chinese Calligraphy Recognition.